The next 6 lines of text, shows us,how a valid configuration file swtorssh.cfg with 6 SSH remote connections should looks like.
Inside of this directory, you find also a simple configuration file for swtorssh.cfg with one host connection. You could use this
sample as a starting point.

```
fullssh.sh  ssh-id Compress 4 2 22  9999 NoShell username1@dnsname.server.de    xxxxx xxxxxx Germany     my-home-server-wan
fullssh.sh  ssh-id Compress 4 2 22  9999 clock   username1@10.0.0.66            xxxxx xxxxxx Germany     my-home-server-lan
fullssh.sh  ssh-id xxxxxxxx 4 2 22  9999 Noshell username2@127.124.166.21       xxxxx backup Austria     freds-server
fullssh.sh  passwd Compress 4 1 443 9999 Noshell hjfdj@84.55.22.166             xxxxx xxxxxx France      french-pgp-server
chainssh.sh ssh-id Compress 4 2 22  9999 chain   username1@dnsname.server.de    11000 xxxxxx Canada      mychain1.cfg
chainssh.sh ssh-id Compress 4 2 22  9999 chain   ujdfhj1@188.67.88.15           11000 xxxxxx Poland      mychain2.cfg
```

This help file describes the exact syntax for the configuration file

~/Persistent/swtor-addon-to-tails/swtorcfg/swtorssh.cfg

This settings are all valid with version 0.52 or higher.

- Every possible ssh-connection is defined in one single line.
- There is no internal limitation of ssh-servers, that can be defined.
- Every ssh server could be defined with a valid ipv4 address or a user friendly dns-name.
- All configuration parameters inside the configuration file swtorssh.cfg are
  case sensitive.
- There are 13 columns possible for every ssh-connection. All 13 fields have to be filled out !
- Spaces “ “ inside the of the fields 12 and 13 are not supported ! Every space “ “ has to be replaced with a “-”.

Column 1:

This column 1 has 3 possible entry's → [fullssh.sh] [chainssh.sh] [pfssh.sh]
Only one of them, can be chosen.

fullssh.sh

This the best possible option for a ssh-connection. In this case we have somewhere a ssh account without any restrictions or
borders for our own actions.

- A login over a password or a login over predefined keys is possible.
- We have enough space to store some backups on that ssh-server.
- We have a unrestricted user shell to use like /bin/bash.
- We could use the commands scp or sshfs to transfer files to the remote ssh-host.
- We could even start a remote X11 applications over the ssh-tunnel like xclock or even a browser.

chainssh.sh

A few ssh-servers on the internet don't like direct any connection attempts made from over the onion-network.
If we would like to connecting to such a onion unfriendly ssh-server, we have to first masquerade our origin
IP with a first ssh server in place,acting as a jump-server. You have to connect to the jump Server
first and from over there without any password to the second ssh-server itself.

tails -> onion-network Nodes 1,2,3 -> ssh-connection > JUMP-SERVER -> ssh-connection -> DESTINATION-SERVER

This scenario is bit a tricky to accomplish,because of the following reasons.

- The desired DESTINATION-SERVER has to support a pass-wordless login.
- There are at least 2 different public/private ssh keys involved.
- Column 11 on the same line contains the second part to use with our ssh-command to the first server.

pfssh.sh

This kind of shell-account is very limited in the usage in general. As a little example, who offer this kind of a limited ssh-access
accounts, you may have a look on this following website.

http://www.dewassh.net/server/SSH

- Only a simple port-forwarding is allowed,nothing more than this simple task can be done.
- No other shell than the good old “jail” called /bin/false is usable over this very limited ssh connection. This means you can only create a remote socks5 proxy,nothing more. You can’t execute any command on the remote server over SSH.
- A login is only possible with a valid password. Logins over a public/private key are not possible or even not desired.
- The ssh-logins has a expiration time between 3 and 30 days. After the expiration of the created account, you have to create a new one.
- Some providers have a large base of country's to choose from where the ssh-server should be came from.
- Some really silly ssh-providers only allow the creation of a ssh-account with a valid Facebook or a Google Email account.
  Of course, we don’t give them away our credentials for Facebook or even a fake Google Email Account.

Column 2:

This column 2 has only 2 possible entry's → [ssh-id] [passwd]
Only one of them, can be chosen.

ssh-id

If we have a private/public key generated for a ssh-connection,we can use this public key stored inside ~/.ssh as a replacement for
a password-less login to the remote server.

passwd

Every time we connect to this ssh-server, we have to provide a valid password.
The password will be send over to the remote ssh-server with the installed sshpass program for tails. The password for this session is not
stored anywhere on the persistent volume. It will be asked in a dialog-box on every startup of the ssh-connection is made.If you are loosing your
password, you can not connect anymore to this server.

Column 3:

This column 3 has 2 possible entry's → [Compress] [xxxxxxxx]
Only one of them, can be chosen.

Compress

By default compression is not enabled for any outgoing ssh connections on any Linux system. Very simple compression is no problem for
modern CPUs and even on a gigabit network it is faster to compress data first before sending it over the network instead of leaving it uncompressed.
This options tells the scripts that we would like to use Compression in our specific used ssh-connection.
On the Linux ssh command-line this would be the option -C.

xxxxxxxx

In the case that no Compression should be used, this dummy text has to be used. But most SSH Servers available, can make use the compression feature,
so it can safely be activated on 99.9 % of all connections.

Column 4:

This column 4 has 1 possible entry → [4]

Tails only supports only TCP-IP Version 4 at all.

Column 5:

This column 5 has 2 possible entry's → [2] [1]
Only one of them, can be chosen.

There are only 2 ssh protocols available to use inside of ssh. Most users should write a 2 here.
Only use the old ssh protocol version 1 ever,if you know exactly, what you are doing !

http://www.snailbook.com/faq/ssh-1-vs-2.auto.html

Column 6:

This column 6 has only one possible entry → [22]

It defines the remote TCP port of the SSH-Server we would like to connecting. The standard port for a ssh server
is 22. This port information may can be something different than 22.

Column 7:

This column 7 has only one possible entry → [9999]

This value describes the port to build our local socks5 server. The only possible value is 9999. You could change this value
if you would like to change it, but this predefined port 9999 is hard encoded into all the scripts. Do change this value
to something other than 9999 only, if you know what you are exactly doing !

Column 8:

This column 8 has 3 possible entry's → [NoShell] [clock] [chain]
Only one of them, can be chosen.

NoShell

After a successful login we could normally start a remote shell on the foreign SSH-Server. Because we are in no need to use
a classic shell after the login this option is used. On a Linux server that shell would be normally /bin/bash.
Please remember to write exactly NoShell and not NOSHELL or Noshell.

clock

Almost every ssh server can make use of the X11-window system. If the statement clock is found here inside the configuration, a nice clock like the following one as example

http://tclock.sourceforge.net/tclock-shot.gif

is displayed as soon we are connected to our ssh server. This option clock needs a fullssh.sh connection inside of the field 01 of the configuration file. The displayed clock now shows the current time of the remote ssh-server.

chain

With this option it is possible to build a "JUMP-SERVER" configuration. This special keyword is mandatory, because we need to connect to a second ssh-server after successful login on the first server itself.

Column 9:

This column 9 has only one possible entry → username1@dnsname.server.com

The data provided here, containing the username and the host-server we would like to connecting over ssh.
A possible entry of this field with a valid DNS server-name:

username1@dnsname.server.com

or a the same entry with a IP instead of a DNS Name :

username1@177.56.43.3

```
----------
Column 10:
----------

This column 10 has 2 possible entry's  → [xxxxx] [11000]
Only one of them, can be chosen.


xxxxxx
------

Almost all users can write this dummy text xxxxx here.


11000
-----

This single port value here written in this field can be in the range of
1024 until 65535.A value here is the only valid in the main mode "chainssh.sh" of field 01,
we need this value as second port definition.


----------
Column 11:
----------

This column 11 has 2 possible entry's  → [xxxxx] [backup]
Only one of them, can be chosen.


xxxxx
-----

Almost all users can write this dummy text xxxxx here.


backup
------

If you have multiple ssh-servers with full shell-access to chose from, with this column 11, you could set exactly one of this ssh-server from the
list as a predefined backup-server. Multiple backup statements inside of the configuration file swtorssh.cfg are not supported by now.
In a later release it will be possible to encrypt the backups as well. By now, the backups created with my scripts are just simply
compressed tar.gz files. You should save this very sensitive backups only to a ssh-host that you can trust at 100 %.
```

Column 12:

This column 12 describe the country, that our used ssh-server is located. Please write a single "-" instead of a single space " " if your complete country-name contains spaces inside of the name.

United Kingdom will be to ->United-Kingdom
Dominican Republic will be -> Dominican-Republic

Of course if we are using the mode “chainssh.sh” we do provide the country where our server2 is located.

Column 13:

This column 13 is a description field for most users with one exception, if you are using chainmode.ssh.
Like in the country field 12 write a single "-" instead of single space " " inside of the text as description.

my-home-server-wan
my-home-server-lan

In the main mode "chainssh.sh" only, we use this description field to create our second connection to the desired SSH-Server. We need to create a new .cfg file for every chainmode.sh connection, we would like to use inside of the add-on. If Columns 13 contains the value mychain1.cfg.
You should create a file called mychain1.cfg inside of the directory ~/Persistent/swtor-addon-to-tails/swtorcfg with a editor of your choice.
In this example we could write the following single command inside this configuration file mychain1.cfg

ssh -AtC42N -D11000 fs-username1@war.freeshells.org

This command will be executed right after logins on our first connected “JUMP-SERVER” to make the desired ssh-connection to the second desired destination server.

- A Enabeshles forwarding of the authentication agent connection
- t Force pseudo-terminal allocation
- C Requests compression of all data
- 4 only use tcp version 4
- 2 Use only ssh 2 protocol
- N Do not execute a remote command. This is useful for just forwarding ports.
- D Specifies a local “dynamic” application-level port forwarding

fs-username1 would be the valid SSH username on the destination server.
war-freeshells.org would be the DNS-Name or the IP Address of the destination server

Note : -D11000 must be made, according the port definition inside the value of column 10.
Numbers below of 1023 can not be used,because only the user root can use this port low numbers below 1023.